UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Dejan Dragman

Analiza rešitev za zagotavljanje visoke razpoložljivosti podatkovnih zbirk strežnika Microsoft SQL Server 2012

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE RAČUNALNIŠTVO IN INFORMATIKA

Ljubljana, 2014

UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Dejan Dragman

Analiza rešitev za zagotavljanje visoke razpoložljivosti podatkovnih zbirk strežnika Microsoft SQL Server 2012

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE RAČUNALNIŠTVO IN INFORMATIKA

MENTOR: viš. pred. dr. Aljaž Zrnec

Ljubljana, 2014

Rezultati diplomskega dela so intelektualna lastnina avtorja in Fakultete za računalništvo in informatiko. Za objavljanje ali izkoriščanje rezultatov diplomskega dela je potrebno pisno soglasje avtorja, Fakultete za računalništvo in informatiko ter mentorja.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Dostopnost in razpoložljivost podatkov je ključni dejavnik za nemoteno delovanje poslovnih procesov. Za zagotavljanje visoke razpoložljivosti podatkovne baze imamo na voljo različne rešitve, ki se med seboj ločijo glede kakovosti, zanesljivosti, hitrosti in ceni. V diplomski nalogi analizirajte več omenjenih rešitev, kjer boste kot kriterije upoštevali: zanesljivost, avtonomnost in hitrost delovanja, varovanje podatkov ter stroške upravljanja in investicije rešitve. Pri tem predstavite posamezne rešitve, njihove prednosti in slabosti, realizirajte njihovo implementacijo ter opravite analizo hitrosti okrevanja in količino izgubljenih podatkov s praktičnim testiranjem posamezne rešitve s simuliranem okrevanjem po katastrofi. Za potrebe praktičnega testiranja razvijte namensko aplikacijo s pomočjo katere boste ugotovili čas okrevanja in količino izgubljenih podatkov za posamezno rešitev.

IZJAVA O AVTORSTVU DIPLOMSKEGA DELA

Spodaj podpisani Dejan Dragman, z vpisno številko **63020203**, sem avtor diplomskega dela z naslovom:

Analiza rešitev za zagotavljanje visoke razpoložljivosti podatkovnih zbirk strežnika Microsoft SQL Server 2012

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom viš. pred. dr. Aljaža Zrneca,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 8. septembra 2014 Podpis avtorja:

Kazalo

Povzetek

Neprekinjen dostop do podatkovnih zbirk ter njihova razpoložljivost odvisnim storitvam in procesom je danes ključni dejavnik v neprekinjenem poslovanju. Pri implementaciji podatkovnega strežnika SQL Server 2012 imamo na voljo različne rešitve za doseganje neprekinjenega poslovanja pri čemer ima vsaka rešitev svoje prednosti in slabosti. Posamezna rešitev v času planiranja odpira vrsto vprašanj o posamezni rešitvi.

Namen diplomskega dela je implementacija in testiranje posamezne rešitve strežnika SQL Server 2012 za zagotavljanje visoke razpoložljivosti podatkovnih zbirk, primerjava rešitev in analiza rezultatov testiranja s pomočjo izbranih kriterijev.

V prvem delu smo predstavili storitve, programska orodja in strežniške rešitve, ki nam omogočajo implementacijo posamezne rešitve. Predstavili smo strežniško gručo, programsko orodje Microsoft SQL Server Management Studio in podatkovni strežnik SQL Server 2012.

V drugem delu smo predstavili posamezno rešitev za zagotavljanje visoke razpoložljivosti podatkovnih zbirk, ki smo jo implementirali in namensko aplikacijo, ki smo jo razvili za izvedbo praktičnega testiranja.

V zadnjem delu smo izvedli praktično testiranje posamezne rešitve, primerjali dobljene rezultate in primerjali posamezne rešitve med sabo.

Ključne besede: podatkovna zbirka, baza, visoka razpoložljivost, neprekinjena razpoložljivost, neprekinjeno delovanje, SQL Server 2012

Abstract

Today's key factor for business continuity is uninterrupted access to databases and their availability to dependent services and processes. Implementation of SQL Server 2012 offers different solution to achieve business continuity, but each solution has its advantages and disadvantages. During the planning phase each solution raises a number of questions.

The purpose of the thesis is to implement and test individual solution of SQL Server 2012 for ensuring high availability of databases, compare solutions and analyse the results of testing based on the selected criteria.

In the first part we presented the services, software tools and server solutions that enabled implementation of individual solution. We presented computer cluster, Microsoft SQL Server Management Studio software tool and SQL Server 2012 database server.

In the second part we presented individual solution for ensuring high availability of databases which we had implemented and a dedicated application that was developed to perform practical testing.

In the last part we carried out a practical test of individual solution, analysed obtained results and compared individual solution between them.

Keywords: database, high availability, continuous availability, continuous operation, SQL Server 2012

1 Uvod

S predstavitvijo računalnika se je svet in življenje močno spremenilo. Z nadaljnjim razvojem računalnika in vpeljavo le tega na različna področja je prišlo do povečane odvisnosti procesov in storitev na področjih kjer je prišlo do vpeljave računalniških sistemov. S tem se je spremenil tudi način hranjenja podatkov. Namesto na listu, ki smo ga shranili v mapo, le to pa vložili v arhivski predalnik smo podatke začeli shranjevati na računalnik. Spremenjena oblika shranjevanja podatkov je omogočila povečevanje količine podatkov, ki jo lahko shranimo in s tem tudi pomembnost le teh. S povečevanjem količine podatkov je prišlo do potrebe po razvoju sistema, ki bi omogočal hitro in učinkovito manipulacijo s podatki pri hranjenju in obdelavi kar je pripeljalo da razvoja sistemov za upravljanje s podatkovnimi zbirkami. Z razvojem sistemov za upravljanje podatkovnih zbirk je postalo upravljanje s podatki v poslovnem svetu, bančništvu, zdravstvu, trgovini in ostalih panogah pomemben dejavnik. S povečevanjem odvisnosti od podatkov, ki se hranijo v podatkovnih zbirkah pa je postala pomembna tudi njihova dostopnost in razpoložljivost.

Informacijska tehnologija v našem življenju danes predstavlja veliko vlogo in smo od nje odvisni v veliki meri. Izpad sistema, ki upravlja s podatkovnimi zbirkami lahko danes predstavlja veliko finančno ali gmotno izgubo, zato je njihova konstantna dostopnost in razpoložljivost izjemno pomembna [5]. Izpad podatkovnega sistema na banki bi nam lahko onemogočil izvedbo bančnih transakcij kot so dvig denarja ali plačilo položnice na bančnem okencu ali v spletni banki. Izpad podatkovnega sistema mobilnega operaterja ali ponudnika internetnega dostopa bi povzročil težave pri komunikaciji od katerih je odvisno delovanje velikega števila sistemov, procesov in ne nazadnje medsebojna komunikacija med posamezniki. Izpad podatkovnega sistema v podjetju lahko onemogoči izdajo materiala iz skladišča, prodajo izdelkov, zastoj proizvodnje in s tem neizkoriščenost zaposlenih v proizvodnji, strojev in naprav. Takšen izpad lahko predstavlja finančno izgubo, konkurenčno izgubo in gmotno škodo v obliki povečane količine odpadnih surovin zaradi zastoja proizvodnega obrata.

V raziskavi podatkovnih centrov [6] v Združenih državah Amerike iz leta 2013 je bilo ugotovljeno, da znaša škoda v povprečju pri vsaki minuti izpada podatkovnega centra okoli 7900 ameriških dolarjev. Kot navaja raziskava se je znesek v primerjavi z letom 2010 povečal

za 41 odstotkov iz tedaj ugotovljenih 5600 ameriških dolarjev. V povprečju je izpad trajal 86 minut in povzročil škodo v vrednosti okoli 690.200 ameriških dolarjev. V letu 2010 je izpad v povprečju trajal 97 minut in povzročil škodo za okoli 505.500 ameriških dolarjev. Izpad spletnega servisa PayPal [7], ki omogoča varno plačevanje v spletu in izmenjavo denarja preko spleta, ki se je zgodil 3. avgusta 2009 in je trajal 4 ure in pol je po ocenah povzročil škodo med 7 in 32 milijoni ameriških dolarjev.

Kot prikazuje raziskava je zanesljivost delovanja podatkovnih centrov pomemben dejavnik pri zanesljivosti poslovanja in omejevanju izgub pri poslovanju. Pri načrtovanju infrastrukture informacijske tehnologije in iz izkušenj uporabnikov storitev se pogosto srečujemo z vprašanji glede ustrezne izbire rešitve, ki bo nudila najboljšo ali pa sprejemljivo mero visoke razpoložljivosti podatkovnih zbirk. V diplomskem delu smo si zadali nalogo, da primerjamo posamezne rešitve, ki jih ponuja strežnik Microsoft SQL Server 2012, preizkusimo njihovo delovanje v praktičnem primeru s testiranjem in nato primerjamo rezultate praktičnih testiranj ter prednosti in slabosti posamezne rešitve. Pri primerjanju posameznih rešitev bomo kot kriterije primerjave upoštevali; zanesljivost, avtonomnost in hitrost delovanja, varovanje podatkov ter stroške upravljanja in investicije rešitve.

2 Programska orodja in storitve

Za izvedbo analize rešitev za zagotavljanje visoke razpoložljivosti podatkovnih zbirk strežnika Microsoft SQL Server 2012 so bila uporabljena sledeča programska orodja in storitve :

- Microsoft SQL Server 2012
- Microsoft SQL Server Management Studio 2012
- Windows Server 2012 R2
- Storitev Failover Cluster

2.1 Microsoft SQL Server 2012

Strežnik Microsoft SQL Server 2012 [2, 21] je skupek programov, ki omogočajo tvorjenje, uporabo in vzdrževanje podatkovnih zbirk.

Strežnik je v osnovi sestavljen iz dveh sistemov za upravljanje podatkovnih zbirk; Database Engine in Analysis Services, storitve Reporting Services, Integration Services in Master Data Services. Vključuje tudi programov in storitve, ki služijo kot podpora pri upravljanju in vzdrževanju podatkovnih zbirk in orodja za upravljanje, ki so nam v pomoč pri upravljanju SQL strežnika, npr. SQL Server Management Studio.

Strežnik Microsoft SQL Server je na voljo v različnih izdajah, ki se delijo na glavne, posebne in razširjene izdaje ter vključujejo različne funkcionalnosti. Tabela 2.1 prikazuje izdaje strežnika Microsoft SQL Server 2012.

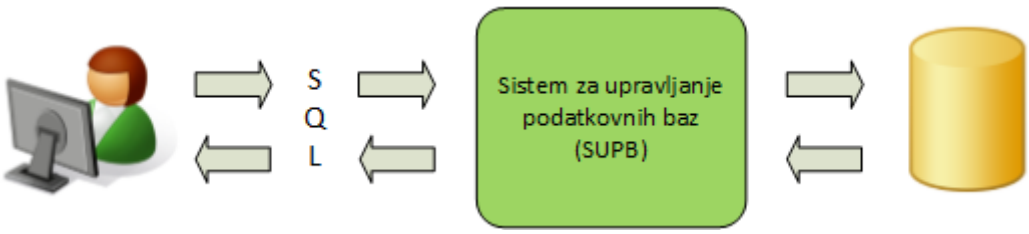
V diplomskem delu smo uporabili izdaji Enterprise in Standard, ki sta poleg izdaje Express, ki je brezplačna najbolj uporabljeni izdaji.

Vrsta izdaje	Ime izdaje
Glavna izdaja	Enterprise
Glavna izdaja	Business Inteligence
Glavna izdaja	Standard
Posebna izdaja	Web
Razširjena izdaja	Developer
Razširjena izdaja	Express

Tabela 2.1: Izdaje strežnika Microsoft SQL Server 2012

2.1.1 Database Engine

Database Engine [2, 21] je sistem za sprotno obdelavo transakcij, ki služi upravljanje podatkovnih zbirk; shranjevanje, procesiranje in varovanje podatkov. Skrbi za upravljanje pri dostopu do podatkov, procesiranju transakcij, ki jih prejema s strani aplikacij, ki želijo dostopati do podatkov z namenom pridobivanja, vpisovanja ali spreminjanja podatkov v podatkovnih zbirkah in skrbi za trajno shranjevanje podatkov. Sestavlja ga pogon za shranjevanje podatkov, ki skrbi za branje in pisanje podatkov na trajni pomnilni medij in povpraševalni procesor, ki razčlenjuje programski jezik Transact-SQL in izvaja njegove ukaze. Uporabnik s pomočjo povpraševanja lahko pridobi podatke iz podatkovne zbirke ali pa jih vanjo shrani oz. jih posodobi. Slika 2.1 prikazuje osnovno delovanje SQL strežnika.

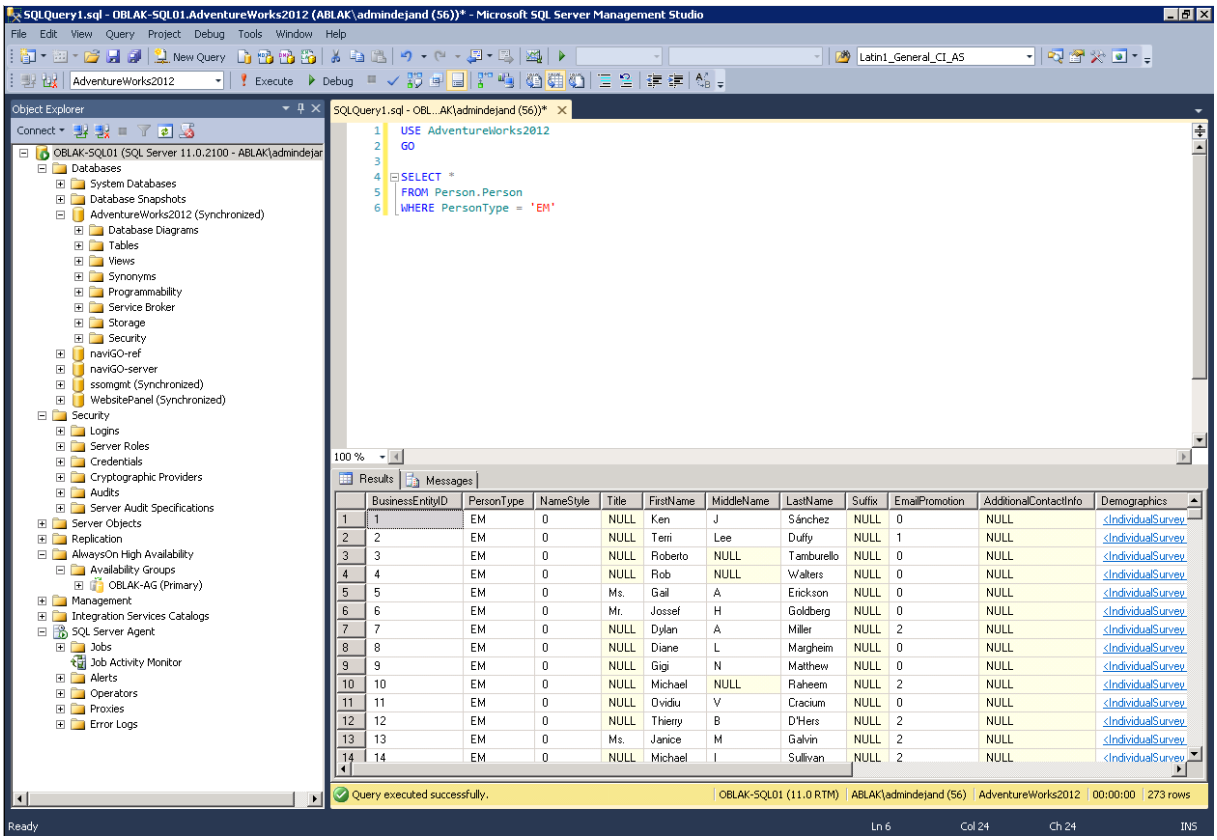


Slika 2.1: Delovanje strežnika SQL

Instanca Database Engine se v operacijskem sistemu izvaja kot kopija izvršilne datoteke sqlserver.exe. Vsak operacijski sistem lahko izvaja več kopij servisa Database Engine, ki ga poimenujemo instanca. Vsaka instanca ima svoje sistemske podatkovne zbirke, ki jih uporablja za delovanje SQL strežnika ter uporabniške podatkovne zbirke kjer so shranjeni podatki.

2.2 Microsoft SQL Server Management Studio 2012

Microsoft SQL Server Management Studio [2] kot prikazuje Slika 2.2 je orodje, ki služi konfiguriranju in upravljanju vseh komponent strežnika SQL Server. Orodje je bilo predstavljeno z izidom strežnika SQL Server 2005 in je del namestitvenega paketa Microsoft SQL Server 2012. Združuje različna grafična in skripta orodja v enovito orodje, ki je v pomoč pri delu s strežnikom in njegovimi komponentami. Najbolj prepoznaven del orodja je Object Explorer, ki uporabniku omogoča interakcijo z objekti strežnika in urejevalnik skript kateri omogoča pisanje, izvajanje in odpravljanje napak v skripti s pomočjo orodja IntelliSense, ki z barvanjem kode v realnem času označuje napake in ponuja predloge med pisanjem skript. Poleg osnovnih orodij vsebuje tudi razširitve, ki omogočajo spremljanje SQL strežnika v realnem času in omogoča grafični izris predvidenega in dejanskega plana izvedbe skript, ki nam je v pomoč pri pisanju in optimizaciji skript. Orodje je mogoče razširiti tudi z lastno razvitimi dodatki oz. dodatki drugih razvijalcev, ki so bodisi brezplačna ali plačljiva.



Slika 2.2: Microsoft SQL Server Management Studio

2.3 Windows Server 2012 R2

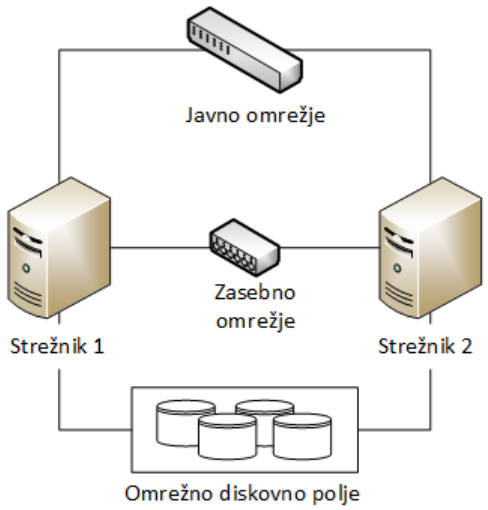
Operacijski sistem Windows Server 2012 R2 [18] je strežniški operacijski sistem, ki je že sedma izdaja operacijskega sistema Windows Server. Skozi razvoj operacijskega sistema je z izdajo novih različic prihajalo do nadgrajevanja storitev in funkcionalnosti, ki jih ponuja operacijski sistem. Zadnja izdaja operacijskega sistema je nadgrajena s storitvami in funkcionalnostmi, ki so pomembne pri virtualizaciji operacijskih sistemov: S svojo nadgradljivostjo in funkcionalno učinkovitostjo omogoča prihranke in učinkovito delovanje v organizacijah vseh velikosti. Z učinkovitim upravljanjem naprav za shranjevanje podatkov prinaša zmogljivost, varnost in prilagodljivost kar je pomemben dejavnik, ki omogoča varno in hitro upravljanje s podatki in se ob tem lahko prilagodi potrebam organizacij različnih velikosti. Uspešno združuje tudi različne omrežne sisteme in naprave, ki omogočajo mrežno povezljivost operacijskega sistema in ob tem ohranja enostavno upravljanje omrežnih sistemov v katerega lahko povežemo večje število strežnikov ter zagotavlja zanesljivo mrežno povezljivost. Slika 2.3 prikazuje najbolj poudarjene funkcionalnosti strežnika, ki so pomembne pri uporabi strežniških sistemov v poslovnem okolju.

Slika 2.3: Funkcionalnosti strežnika Windows Server 2012 R2 [10, 18]

Dodatne funkcionalnosti in razširitve, ki jih lahko namestimo na zahtevo, omogočajo upravljanje strežniške infrastrukture in avtomatizacijo kjer se zaradi obsežnosti pokaže potreba po centraliziranem upravljanju. Operacijski sistem omogoča tudi nadgradnjo s spletnim strežnikom, ki omogoča uporabo spletnih storitev v lokalnem ali javnem okolju, ki v kombinaciji z aktivnim imenikom, federacijo aktivnega imenika in dinamično kontrolo dostopa omogoča poenostavljen dostop do povezanih vendar med seboj neodvisnih storitev v organizaciji s poenoteno identiteto uporabnika. Poudarek operacijskega sistema na storitvi virtualizacije in povezava med seboj neodvisnih storitev omogoča vzpostavitev okolja virtualiziranih delovnih postaj, ki zagotavlja celovit sistem za upravljanje delovnih postaj.

2.4 Storitev Failover Cluster

Failover Cluster oz. strežniška gruča [1, 2, 4, 9, 12, 13, 14, 15] je funkcija operacijskega sistema Windows Server, ki povezuje strežnike v gručo in zagotavlja visoko zanesljivost delovanja aplikacij in storitev. Prva različica strežniške gruče je bila predstavljena z izidom operacijskega sistema Windows NT Server 4.0 leta 1996. Z izidom novejših različic operacijskega sistema Windows Server se je razvijala tudi funkcija strežniške gruče. Leta 2006 je Microsoft predstavil tudi tehnologijo strežniške gruče za porazdeljene računske operacije z namenom povečevanja procesorske in računske moči z imenom Windows Compute Cluster Server 2003 [16], ki ga redno posodablja z izdajo nove različice operacijskega sistema Windows Server.



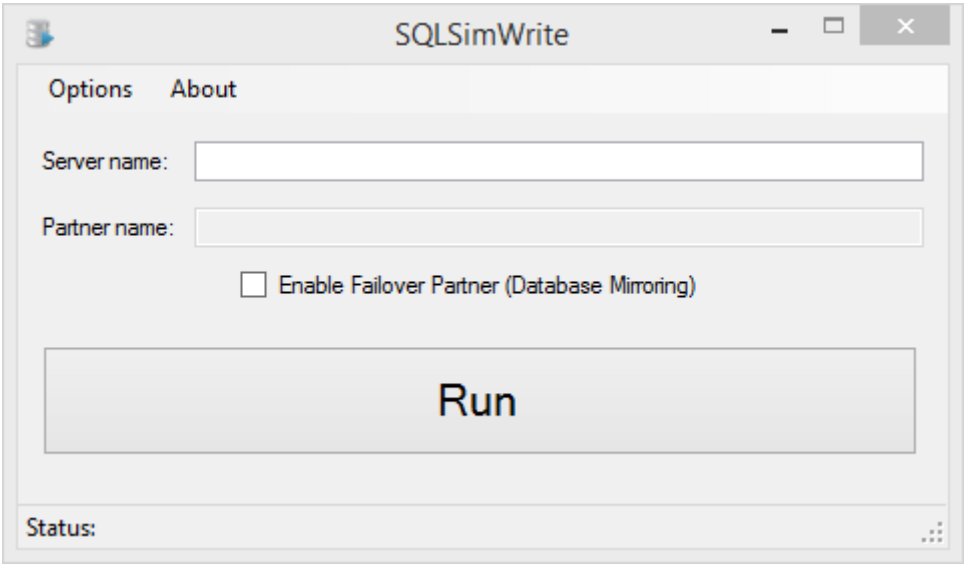
Slika 2.4: Strežniška gruča

Strežniška gruča (Slika 2.4) med sabo povezuje dva ali več neodvisnih strežnikov, ki so vključeni v strežniško gručo. Strežniki v gruči so med sabo povezani z mrežno povezavo, ki jo označujemo kot javna mrežna povezava. Po njej se prenašajo podatki, ki so potrebni za komunikacijo med strežniki in odjemalci ter v določenih primerih tudi komunikacija med strežniki, ki so vključeni v strežniško gručo. Ker je komunikacija med strežniki, ki so del strežniške gruče, pomembna jih med sabo povezujemo z dodatno, zasebno mrežno povezavo (heartbeat) po kateri se prenaša komunikacija med strežniki v strežniški gruči. Pomemben del strežniške gruče, vendar ne vedno obvezen del, kot predstavljeno v nadaljevanju naloge, je omrežno diskovno polje (SAN), ki služi shranjevanju podatkov. Omrežno diskovno polje je strežniški sistem v katerem se nahaja nabor trdih diskov, do katerih lahko dostopajo vsi strežniki, ki so del strežniške gruče. Pri vzpostavitvi Microsoft SQL Server strežniške gruče je omrežno diskovno polje potrebna strojna oprema, ker so na njem shranjene podatkovne

zbirke. Vzpostavitev določenih storitev v strežniški gruči ne potrebuje omrežnega diskovnega polja, ker se podatki shranjujejo na lokalne diske, ki so del strežnika kateri je del strežniške gruče. Storitev Always On Availability Group je primer takšne storitve, ki je ena izmed mehanizmov za zagotavljanje visoke zanesljivosti podatkovnih zbirk strežnika Microsoft SQL Server 2012. Pri odpovedi delovanja strežnika, ki je del strežniške gruče se delovanje aplikacij in storitev, ki so delovale na strežniku na katerem je prišlo do odpovedi, nadaljuje na drugem strežniku, ki lahko prevzame delovanje aplikacije in storitve.

2.5 Aplikacija SQLSimWrite

Aplikacijo SQLSimWrite (Slika 2.5) smo razvili v okviru potreb diplomskega dela za testiranje delovanja rešitev, ki zagotavljajo visoko zanesljivost. Aplikacija vpisuje čas izvedbe transakcije v testno podatkovno zbirko strežnika Microsoft SQL Server. V primeru izgube povezave s strežnikom, aplikacija poskuša vzpostaviti povezavo z neskončnim številom poskusov ali do zaustavitve aplikacije. Aplikacija podpira delovanje z vsemi rešitvami, ki smo jih predstavili v diplomskem delu, poleg vpisovanja transakcij v podatkovne zbirke pa omogoča tudi praznjenje tabele testne podatkovne zbirke. Testna podatkovna zbirka in tabela v podatkovni zbirki je definirana v aplikaciji z namenom razločnosti namena testne podatkovne zbirke. Za izdelavo aplikacije smo uporabili programsko orodje Microsoft Visual Studio 2012 [19], programski jezik C# [8, 11] in Microsoft .NET ogrodje [11].



Slika 2.5:Aplikacija SQLSimWrite

3 Rešitve visoke razpoložljivosti

Strežnik SQL Server 2012 omogoča implementacijo štirih rešitev za zagotavljanje visoke razpoložljivosti [9, 14, 15]. Delovanje predstavljenih rešitev lahko razdelimo z vidika nivoja in z vidika načina delovanja [3]. Z vidika nivoja delovanja rešitev lahko rešitve ločimo na delovanje na nivoju celotne instance strežnika SQL kjer rešitev upravlja z delovanjem celotne instance strežnika SQL in na delovanje na nivoju posamezne podatkovne zbirke kjer rešitev upravlja samo s podatkovno zbirko, ki je vključena v rešitev oz. s skupino podatkovnih zbirk, ki so vključene v rešitev. Z vidika načina delovanja pa rešitve lahko delimo na delovanje z deljenjem podatkovnih zbirk med strežniki in z repliciranje podatkovnih zbirk med strežniki, ki so vključeni v rešitev. Pri delovanju z deljenem, strežniki ki so vključeni v rešitev s podporo strežniške gruče deljeno dostopajo do podatkovnih zbirk glede na dodeljeno pravico za dostop. Pri delovanju z repliciranjem vsak strežnik hrani svojo repliko oz. kopijo podatkovne zbirke, ki jo z največjo mero učinkovitosti posodablja s spremembami, ki se dogajajo na podatkovni zbirki, ki je primarna podatkovna zbirka do katere dostopajo odjemalci ter iz nje berejo podatke in vanjo zapisujejo in v njej spreminjajo podatke.

Predstavljene rešitve za svoje delovanje implementirajo in uporabljajo različne storitve in servise, pri čemer pa je moč opaziti podobnost pri delovanju določenih rešitev. Podobnost pri delovanju je odraz nadgrajevanja funkcionalnosti strežnika Microsoft SQL 2012 skozi razvoj strežnika in implementiranje rešitev iz drugih strežniških produktov podjetje Microsoft. Primer takšne implementacije je Microsoft Exchange 2010 Database Availability Group [16], ki ga je podjetje Microsoft implementiralo v strežnik Microsoft SQL Server 2012 pri razvoju rešitve SQL Server AlwaysOn Availability Group (skupina za razpoložljivost).

3.1 Strežniška gruča

Strežniška gruča oz. AlwaysOn Failover Cluster Instances [1, 2, 9, 14, 15] je skupina neodvisnih med seboj povezanih strežnikov s pomočjo mrežne in programske opreme z namenom povečevanja razpoložljivosti storitev, ki so vključene v strežniško gručo. Strežniki v gruči so nosilci storitev strežniške gruče (instance SQL strežnika), katere se s strani strežniške gruče aktivno nadzirajo. Pri izpadu delovanja storitve strežniške gruče se le ta ponovno zažene oz. se storitev prestavi na drug strežnik v strežniški gruči. SQL strežniška

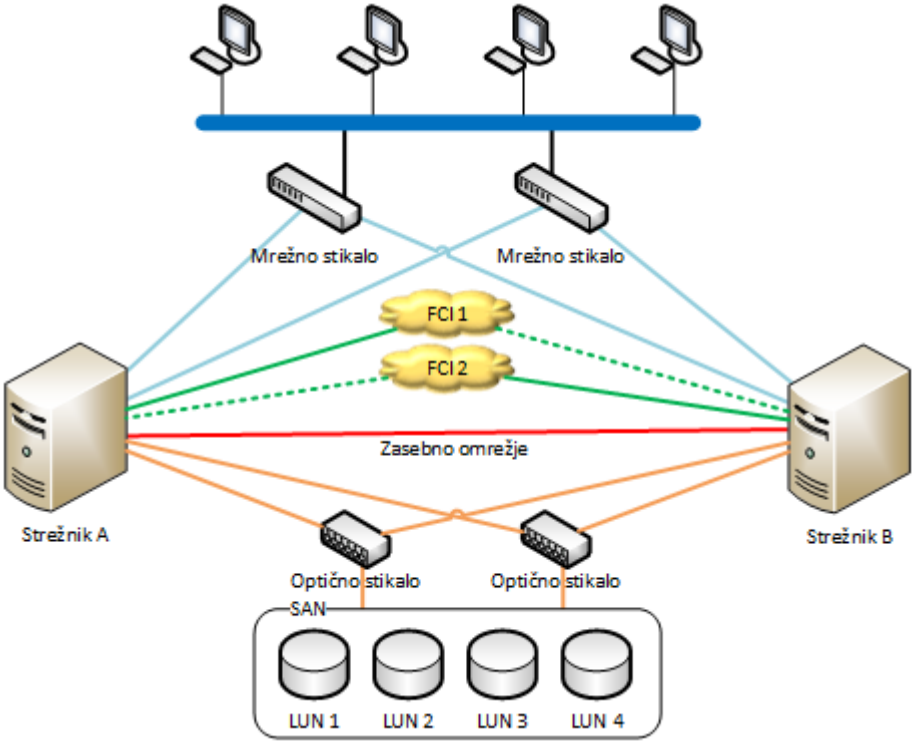
gruča za svoje delovanje zahteva tudi omrežno diskovno polje (SAN) [1, 3]. Omrežno diskovno polje vsebuje eno ali več logičnih diskovnih enot (LUN) [1], ki jih sestavlja eden ali več trdih diskov, ki so povezani v logično diskovno enoto, ki jo strežnik vidi kot en trdi disk kjer so shranjene podatkovne zbirke. V strežniško gručo lahko vključimo en ali več strežnikov, vendar je priporočljivo v gručo vključiti najmanj dva strežnika, da zagotovimo visoko zanesljivost storitev strežniške gruče. Prednosti rešitve so robustnost in visoka zanesljivost, čas izpada delovanja strežnika SQL zaradi napak ali vzdrževanja je majhen zaradi avtonomnega in avtomatiziranega delovanja rešitve in enostavno upravljanje. Slaba stran rešitve je zahtevnost implementacije, potrebuje omrežje diskovno polje in ne varuje podatkov pred okvaro, nenamernemu izbrisu ali vnosu napačnega podatka.

Vsaka instanca strežnika SQL (FCI) v strežniški gruči predstavlja samostojno, neodvisno in nepovezano storitev z ostalimi storitvami strežniške gruče. Pri namestitvi instance strežnika SQL v strežniško gručo na vsakega izmed strežnikov v strežniški gruči namestimo potrebne servise za določeno instanco strežnika SQL, ki jih upravlja strežniška gruča skupaj z ostalimi viri instance, ki jih določimo ob namestitvi instance v strežniško gručo.

Viri instance so:

- Navidezno mrežno ime
- IP naslov
- Logične diskovne enote
- Servis Database Engine
- Servis Server Agent

Viri instance so aktivni na strežniku kjer je aktivna instanca, na ostalih strežnikih pa so zaustavljeni. Strežnik, ki gosti instanco ima v času delovanja instance lastništvo nad viri instance. Če pride do izpada delovanja instance na strežniku jo strežniška gruča ponovno zažene. Če je poskus zagona neuspešen se lastništvo virov instance prenese na drug strežnik v strežniški gruči, ki zažene vire instance. Poenostavljeno lahko rečemo, da je instanca v strežniški gruči v skupni rabi in do nje lahko dostopijo vsi strežniki, ki jo lahko gostijo. Pri tem ohrani vse svoje lastnosti, ker gre za eno kopijo podatkov, ki je shranjena na logičnih diskovnih enotah do katere lahko dostopijo vsi strežniki, ki lahko gostijo instanco. Slika 3.1 prikazuje primer strežniške gruče z dvema strežnikoma v gruči, dvema instancama strežnika SQL in štirimi logičnimi diskovnimi enotami. Slika prikazuje tudi podvojeno povezavo med strežnikoma in SAN diskovnim poljem s pomočjo optičnih stikal, ki zagotavlja potrebno redundanco na nivoju strojne opreme. Redundanca je zagotovljena tudi s povezavo obeh strežnikov v računalniško mrežo s pomočjo dveh mrežnih stikal.



Slika 3.1: Strežniška gruča z instancama strežnika SQL

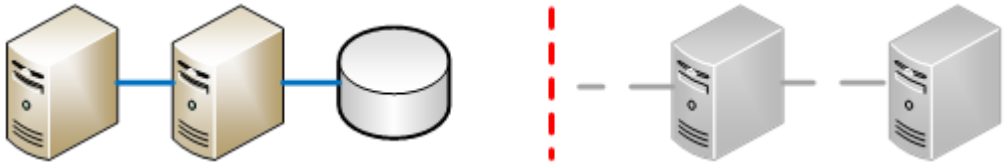
V prikazanem primeru (Slika 3.1) je instanca FCI 1 aktivna na strežniku A, instanca FCI 2 pa je aktivna na strežniku B. Če pride do izpada delovanja strežnika B in s tem instance FCI 2, strežniška gruča zazna izpad delovanja strežnika in instance na strežniku ter prenese pravice za vire instance na strežnik A. Strežnik A z dodeljenimi pravicami zažene servise instance FCI 2, ki dostopi do podatkov, ki so shranjeni na logični diskovni enoti instance in zažene delovanje strežnika SQL. Uporabniki, ki dostopajo do instance FCI 2 lahko v tem primeru zaznajo krajši izpad delovanja strežnika SQL. Čas izpada, ki se zgodi zaradi prenosa virov instance med strežniki je odvisen od količine podatkov, ki jim mora Database Engine obdelati pri procesu obnovitve podatkovnih zbirk v skladno stanje.

Za pravilno in hitro delovanje preklopa storitev v strežniški gruči, strežniška gruča tvori in ohranja kvorum [1, 12, 13, 14, 15] in komunikacijo z aktivno instanco strežnika SQL, ki je registrirana v strežniški gruči, da preverja stanje instance. Konfiguracija kvoruma v strežniški gruči določa največje število napak strežniške gruče, ki še dovoljuje njeno varno delovanje in deluje po principu glasovanja glede na konfiguracijo kvoruma. Odsotnost kvoruma nakazuje na nezanesljivo stanje strežniške gruče, ki vodi v zaustavitev storitev strežniške gruče do ponovne vzpostavitve kvoruma.

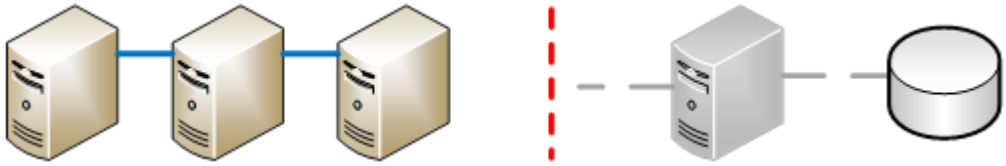
Strežniška gruča omogoča štiri načine tvorjenja kvoruma, ki opredelijo upravljanje strežniške gruče pri glasovanju posameznih virov kvoruma, ki so del strežniške gruče;

- Večina strežnikov (Node Majority)
- Večina strežnikov in disk (Node and Disk Majority)
- Večina strežnikov in datoteka v skupni rabi (Node and File Share Majority)
- Samo disk (Disk Only)

V prikazanem primeru postavitve strežniške gruče (Slika 3.1) smo uporabili konfiguracijo kvoruma z večino strežnikov in disk. Takšna konfiguracija dodeljuje posameznemu strežniku, ki je del strežniške gruče en glas pri glasovanju za vzdrževanje kvoruma in en glas disku, ki se nahaja na SAN diskovnem polju kot logična diskovna enota. Delovanje strežniške gruče je zanesljivo, če je delujočih polovica strežnikov v strežniški gruči in disk kvoruma (Slika 3.2) ali če deluje več kot polovica strežnikov (Slika 3.3).



Slika 3.2: Kvorum z diskom v kvorumu



Slika 3.3: Kvorum brez diska v kvorumu

Aktivna instanca strežnika SQL v strežniški gruči neprestano komunicira s strežniško gručo, ki definira pogoje za ponoven zagon storitev in strežnikov ali prenosa storitev strežniške gruče na drug strežnik. Definirane pogoje delimo na tri dele;

- Spremljanje stanja instance
- Ugotavljanje napak
- Odziv na napake

Strežniška gruča pri spremljanju stanja instance nadzira stanje delovanja instance, njeno odzivnost in zbira diagnostične podatke o instanci s pomočjo katerih lahko zazna napako pri

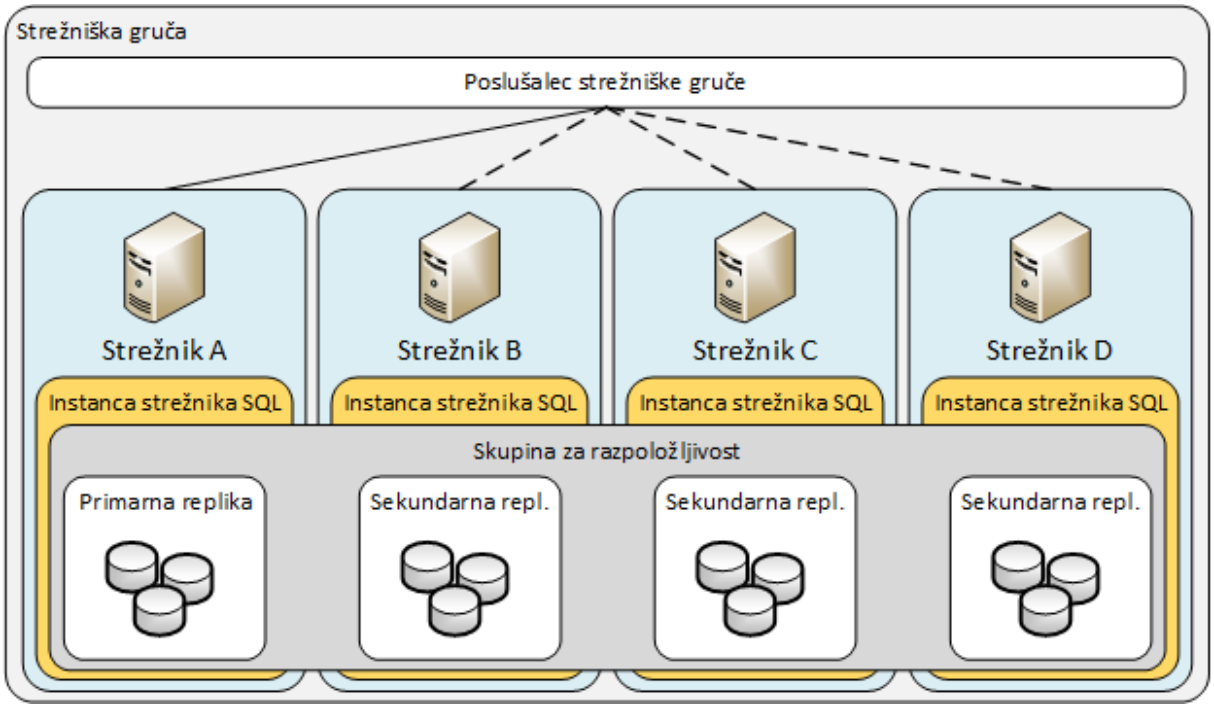
delovanju instance. Pri ugotavljanju napake se s pomočjo ugotovljenega stanja instance določi ali gre za napako in na podlagi stopnje ugotovljene napake se določi kakšen bo odziv na napako. Pri odzivu na napako se strežniška gruča odziva glede na stanje kvoruma in glede na nastavitve sredstev v strežniški gruči.

V okviru diplomskega dela smo postavili rešitev strežniške gruče z eno instanco strežnika SQL (FCI1) kot je prikazana na Slika 3.1. Pri tem smo uporabili dva strežnika, kot je prikazano na sliki, uporabili redundantno povezavo med strežnikoma in omrežnim diskovnim poljem ter med strežnikoma in mrežnimi stikali. Strežnik A je aktivni gostitelj instance strežnika SQL, strežnik B pa je neaktivni gostitelj instance, ki čaka na prevzem instance pri napaki v delovanju strežnika A. Na instanci FCI1 se nahaja podatkovna zbirka TestDB, ki smo jo uporabili v preizkusu visoke zanesljivosti. Preizkus visoke zanesljivosti rešitev smo preizkusili na dva načina. Pri prvem testu smo simulirano odklopili strežnik A s prekinitvijo vseh povezav na strežniku A (optične in mrežne povezave) in tako poustvarili nepredviden izpad strežnika. V drugem testu smo preko konzole za upravljanje strežniške gruče opravili predviden oz. nadzorovan izpad storitve strežniške gruče z ročnim prenosom instance FCI1 iz strežnika A na strežnik B. V obeh simulacijah smo beležili čas izpada podatkovnega strežnika in količino izgubljenih podatkov.

3.2 Skupina za razpoložljivost

Skupina za razpoložljivost oz. AlwaysOn Availability Group [2, 9, 14, 15] je rešitev za zagotavljanje visoke zanesljivosti, ki je nadgradnja rešitve zrcaljenje podatkovnih zbirk. Rešitev za svoje delovanje potrebuje vzpostavitev strežniške gruče katera služi upravljanju vseh strežnikov, ki so vključeni v rešitev in zagotavljanja poslušalca, ki služi kot enotna točka za povezavo odjemalcev na skupino za razpoložljivost. Poslušalec strežniške gruče je računalniški objekt, ki je shranjen v aktivnega imeniku in ima svoje domensko ime na domenskem strežniku kamor zapisuje podatke za dostop do primarne replike, ki jih odjemalci potrebujejo. Pri preklopu kjer se zamenjajo vloge primarne in sekundarne replike, sekundarna replika postane primarna replika, do tedaj primarna replika pa postane sekundarna replika. Sprememba o vlogah replike se zapiše tudi v računalniški objekt poslušalca v domenskem strežniku. Skupino za razpoložljivost lahko sestavlja do 5 replik pri čemer je ena replika primarna kjer je aktivna oz. produkcijska podatkovna zbirka in 4 sekundarne replike, ki vsebujejo kopijo podatkovne zbirke in omogočajo dostop do kopije podatkovne zbirke za branje in izvajanje varnostnega kopiranja. Vsak strežnik, ki je vključen v rešitev predstavlja eno repliko in gosti eno ali več instanc strežnika SQL. Instanca lahko gosti eno ali več skupin za razpoložljivost, katera lahko gosti več podatkovnih zbirk. Podatkovne zbirke so shranjene na lokalnem diskovnem polju strežnika, zato za vzpostavitev rešitve ne potrebujemo omrežnega diskovnega polja (SAN). Skupina za razpoložljivost deluje na nivoju podatkovne zbirke, vendar lahko v eno zbirko vključimo več podatkovnih zbirk in tako v okviru ene skupine upravljamo več podatkovnih zbirk.

Skupina za razpoložljivost pri sinhronem načinu delovanja s samodejnim preklopom za pravilno in hitro preklapljanje vlog med primarnimi in sekundarnimi replikami tvori in ohranja kvorum, ki ga sestavljajo vsi strežniki v strežniški gruči, tudi če ne gostijo replike skupine za razpoložljivost. Ker rešitev za delovanje ne potrebuje omrežnega diskovnega polja, lahko kvorum, ki smo ga opisali v poglavju Strežniška gruča, konfiguriramo v načinu Večina strežnikov ali Večina strežnikov in datoteka v skupni rabi. Ker je konfiguracija kvoruma Večina strežnikov in datoteka v skupni rabi bolj robustna kot Večina strežnikov uporabljamo prvo in datoteko v skupni rabi vzpostavimo na strežniku, ki ne gosti replike skupine za razpoložljivost in ni vključen v strežniško gručo.



Slika 3.4: Skupina za visoko razpoložljivost

Slika 3.4 prikazuje primer postavitve skupine za visoko razpoložljivost s poslušalcem, kjer vsak strežnik gosti instanco strežnika SQL. SQL instance na posameznih strežnikih gostijo skupino za razpoložljivost (Skupina za razpoložljivost), ki gosti tri podatkovne zbirke s primarno repliko na strežniku A in tremi sekundarnimi replikami na strežnikih B, C in D.

Rešitev omogoča dva načina delovanja:

- Asinhron način (Asynchronous-commit mode)
- Sinhron način (Synchronous-commit mode)

Asinhron način delovanja ne zahteva identičnega stanja med primarno in sekundarno repliko podatkovne zbirke, ko pride do spremembe v primarni repliki. Pri sprejeti transakciji primarna replika zapiše spremembo v transakcijski dnevnik, replikam pošlje aktivni del transakcijskega dnevnika, ki vsebuje spremembe, ki so se zgodile na primarni repliki in odjemalcu sporoči, da je bila transakcija uspešno dokončana. Primarna replika, ne glede na nastavljen način sinhronizacije na sekundarni repliki, pri tem ne čaka na povratno informacijo o uspešnem vnosu sprememb na sekundarni repliki, zato sekundarna replika vedno zaostaja za primarno repliko. Zaostanek sekundarne replike je odvisen od hitrosti povezave med primarno in sekundarno repliko in od obremenitve sekundarne replike. Tovrsten način delovanja omogoča hitro delovanje brez zakasnitev transakcij na daljši razdalji med replikami in ker daje prednost

hitrosti nad varovanjem podatkov je način delovanja uporaben za zagotavljanje sekundarne replike, ki jo lahko uporabimo v primeru okrevanja po katastrofi. Delovanje v asinhronem načinu omogoča prisiljen preklop med replikami, ki zamenja vlogo delovanja med primarno in sekundarno repliko. Pri prisiljenem preklopu lahko pride do izgube podatkov na sekundarni repliki, količina izgubljenih podatkov pa je odvisna od zaostanka sekundarne replike za primarno repliko in od sprememb na primarni repliki katerih primarna replika še ni posredovala sekundarni repliki iz aktivnega transakcijskega dnevnika. Asinhron način delovanja zaradi zaostajanja sekundarne replike ne omogoča samodejnega preklopa vlog med primarno in sekundarno repliko, če pride do izpada primarne replike. Če želimo sekundarno repliko postaviti v vlogo primarne replike pri izpadu prvotne primarne replike je potrebno izvesti prisiljen preklop kjer lahko pride do že omenjene izgube podatkov.

Sinhron način delovanja zahteva identično stanje v vseh replikah podatkovne zbirke v vsakem trenutku preden lahko nadaljujemo z delom, zagotavlja preklop med primarno in sekundarno repliko brez izgube podatkov in nam tako omogoča visoko varnost varovanja podatkov. Ko primarna replika sprejme transakcijo, ki spremeni podatke v podatkovni zbirki jo zapiše v transakcijski dnevnik in pošlje vsebino aktivnega dela transakcijskega dnevnika sekundarni repliki in pri ter ohrani transakcijo aktivno. Sekundarna replika prejeti aktivni del transakcijskega dnevnika zapiše v svoj transakcijski dnevnik in ko konča z vpisom pošlje primarni repliki potrdilo o uspešnem vpisu. Ko primarna replika prejme potrdilo o uspešnem vpisu sprememb na sekundarni repliki, in s tem tudi potrdilo o sinhroniziranem stanju sekundarne replike s primarno repliko, zaključi transakcijo in odjemalcu sporoči, da je bila transakcijo uspešno izvedena.

Sinhrono delovanje se vzdržuje z naslednjimi koraki:

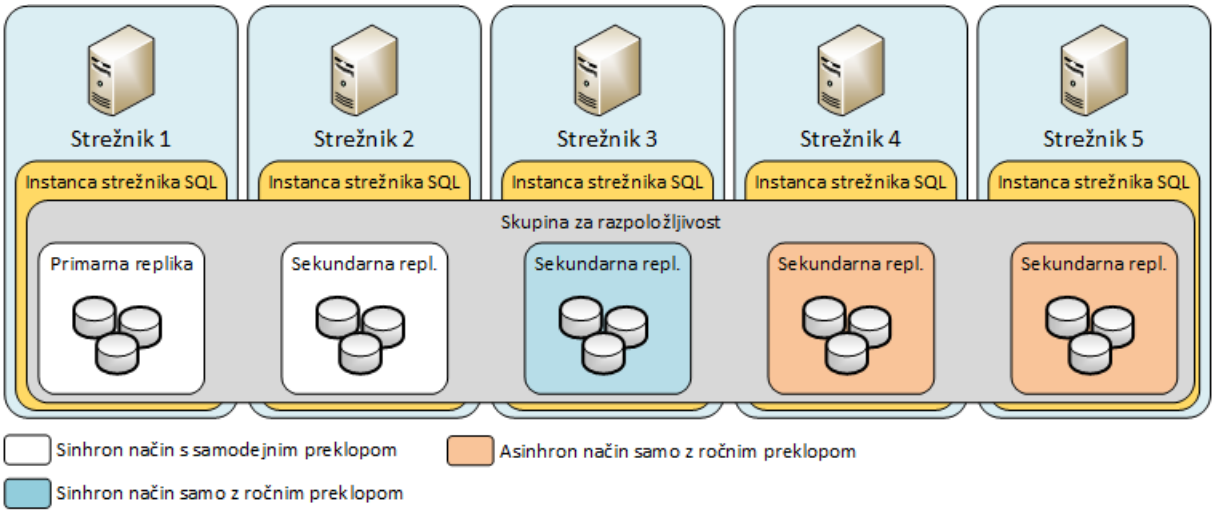
1. Primarni strežnik transakcijo ob prejemu od odjemalca zapiše v transakcijski dnevnik, pošlje aktivni del transakcijskega dnevnika sekundarni repliki in ohrani transakcijo aktivno
2. Primarna replika čaka na potrditev iz sekundarne replike o uspešnem vpisu spremembe v transakcijski dnevnik
3. Sekundarna replika zapiše spremembo v transakcijski dnevnik replike in primarni repliki pošlje potrdilo o uspešnem vpisu sprememb
4. Primarna replika po prejetju potrdila s strani sekundarne replike zaključi transakcijo in odjemalcu sporoči, da je bila transakcija uspešno izvedena

Zaradi čakanja na potrditev o uspešnem vpisu sprememb v sekundarni repliki podatkovne zbirke prihaja pri sinhron načinu delovanja skupine za razpoložljivost do zakasnitev

transakcij. Zakasnitev se pri odjemalcih odraža kot čakanje na uspešno dokončanje transakcije. Dejavniki, ki vplivajo na hitrost sinhronizacije so obremenjenost sekundarne replike in hitrost povezave med replikama. Sekundarna replika ima določen čas v katerem mora potrditi uspešen vpis sprememb. Če pride do izteka časa za potrditev, primarna replika označi sekundarno repliko kot okvarjeno, stanje povezave s sekundarno repliko spremeni v prekinjeno in preneha s čakanjem na potrditve s strani sekundarne replike. Tovrstno obnašanje pri sinhronem načinu delovanja preprečuje zastoje pri zaključevanju transakcij na primarni repliki v primeru okvare sekundarne replike.

Sinhron način delovanja omogoča ročni in samodejni preklop med primarno in sekundarno repliko. Ročni preklo se izvede s posredovanjem skrbnika strežnika in rešitve, kjer zaradi sinhronega načina delovanja ne pride do izgube podatkov. Če se ročni preklop izvede v primeru izpada primarne replike lahko pride do izgube podatkov, če so bile spremembe, ki so bile poslane iz primarne replike na sekundarno repliko izgubljene zaradi npr. izgube mrežne povezave med replikama. Pri samodejnem preklopu se priklop izvede s strani strežniške gruče, če pride do izpada primarne replike, če so vse podatkovne zbirke sinhronizirane in ima skupina za razpoložljivost pravilno konfiguriran in delujoč kvorum. Za vklop samodejnega preklopa moramo primarno in sekundarno repliko nastaviti na sinhron način delovanja ter pri obeh vklopiti samodejni preklop.

Vsaki repliki, ki je del skupine za razpoložljivost ločeno določimo način v katerem bo replika delovala. Slika 3.5 prikazuje primer različnih načinov delovanja med primarno in sekundarnimi replikami. Repliki na strežniku 1 in 2 delujeta v sinhronem načinu delovanja s samodejnim preklopom, replika na strežnik 3 deluje v sinhronem načinu z ročnim preklopom, repliki na strežniku 4 in 5 pa delujeta v asinhronem načinu, ki omogoča samo prisiljen preklop.



Slika 3.5: Različni načini delovanja med replikami

Tabela 3.1 prikazuje povzetek preklopov med replikami in načine delovanja med replikami glede na repliko strežnika, ki ima trenutno vlogo primarne replike.

Primarna replika	**Samodejen preklop**	**Sinhron način delovanja**	**Asinhron način delovanja**	**Samodejen preklop**
Strežnik 1	Strežnik 2	Strežnik 2 in 3	Strežnik 4	DA
Strežnik 2	Strežnik 1	Strežnik 1 in 3	Strežnik 4	DA
Strežnik 3		Strežnik 1 n 2	Strežnik 4	NE
Strežnik 4			Strežnik 1, 2 in 3	NE

Tabela 3.1: Način delovanja med replikami

V okviru diplomskega dela smo za postavitev rešitve skupine za razpoložljivost uporabili dva strežnika (TEST-SQL01 in TEST-SQL02), ki imata vzpostavljeno medsebojno povezavo z lokalno mrežo. Strežnik TEST-SQL01 služi kot primarna replika kjer se nahaja podatkovna zbirka TestDB, ki smo jo uporabili v preizkusu. Strežnik TEST-SQL02 služi kot sekundarna replika kjer se nahaja kopija podatkovne zbirke TestDB. Pri testiranju rešitve visoke zanesljivosti smo izvedli dva načina testiranja v sinhronem načinu delovanja s samodejnim preklopom. Pri prvem testu smo simulirali preklop med primarno in sekundarno repliko. V drugem primeru smo izvedli nepričakovano odpoved primarne replike TEST-SQL01. V obeh simulacijah smo beležili čas izpada podatkovnega strežnika in količino izgubljenih podatkov.

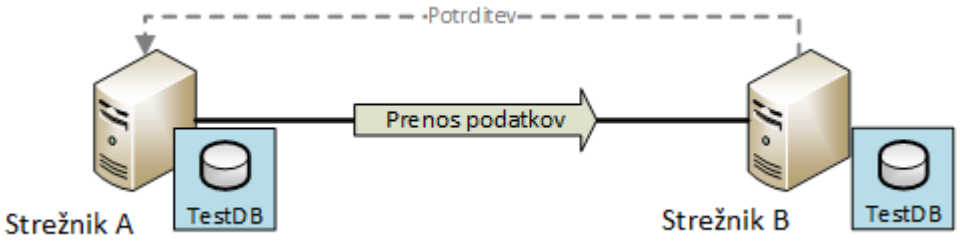
3.3 Zrcaljenje podatkovne zbirke

Zrcaljenje podatkovne zbirke oz. Database mirroring [1, 14, 15] vzdržuje dve kopiji podatkovne zbirke, ki se nahajata na dveh različnih strežnikih. Prva kopija, ki je tudi aktivna, produkcijska kopija do katere dostopajo odjemalci se nahaja na primarnem strežniku, druga, zrcalna kopija pa se nahaja na zrcalnem strežniku. Podatki oz. spremembe, ki se zgodijo v podatkovni zbirki na primarnem strežniku se replicirajo na zrcalno kopijo na zrcalnem strežniku. Rešitev deluje na nivoju podatkovne zbirke, za svoje delovanje pa potrebuje dva strežnika na katerih je aktivna instanca strežnika SQL in opcijsko dodaten, nadzorni strežnik z aktivno instanco strežnika SQL. Potreba po nadzornem strežniku je odvisna od konfiguracije zrcaljenja. Odjemalci povezavo med primarnim in zrcalnim strežnikom vzpostavljajo s pomočjo povezovalnega niza v katerega vpišejo tudi zrcalni strežnik na katerega se odjemalec poveže v primeru nedosegljivosti primarnega strežnika.

Rešitev omogoča dva načina delovanja

- Visoka varnost (High-Safety Mode)
- Visoka zmogljivost (High-Performance Mode)

Način delovanja rešitve z visoko varnostjo (Slika 3.6) deluje v sinhronem načinu, ki zahteva identično stanje v primarni in zrcalni podatkovni zbirki preden lahko nadaljujemo z delom in s tem zagotavlja preklop med strežniki brez izgube podatkov. Ko pride do transakcije, ki spremeni podatke v podatkovni zbirki na primarnem strežniku se transakcija zapiše v transakcijski dnevnik, aktivna vsebina transakcijskega dnevnika, ki je povzročila spremembe se pošlje na zrcalni strežnik in ohrani transakcijo aktivno. Zrcalni strežnik vse spremembe zapiše v zrcalno podatkovno zbirko in s tem zagotovi sinhrono stanje med primarno in zrcalno podatkovno zbirko. Transakcija je na obeh strežnikih dokončana, ko zrcalni strežnik potrdi vnos sprememb primarnemu strežniku kateri zaključi transakcijo. Ko se vzpostavi sinhrono stanje med podatkovnima zbirkama lahko nadaljujemo z delom na primarni podatkovni zbirki. Zaradi čakanja na potrditev iz zrcalnega strežnika prihaja do zakasnitve pri transakcijah, ki se pri odjemalcih izkaže kot povečan čas čakanja. Zakasnitev je odvisna od zaostanka zrcalne podatkovne zbirke za primarno podatkovno zbirko, obremenitve primarne podatkovne zbirke, hitrosti zrcalnega strežnika in hitrosti mrežne ali optične povezave med strežnikoma.

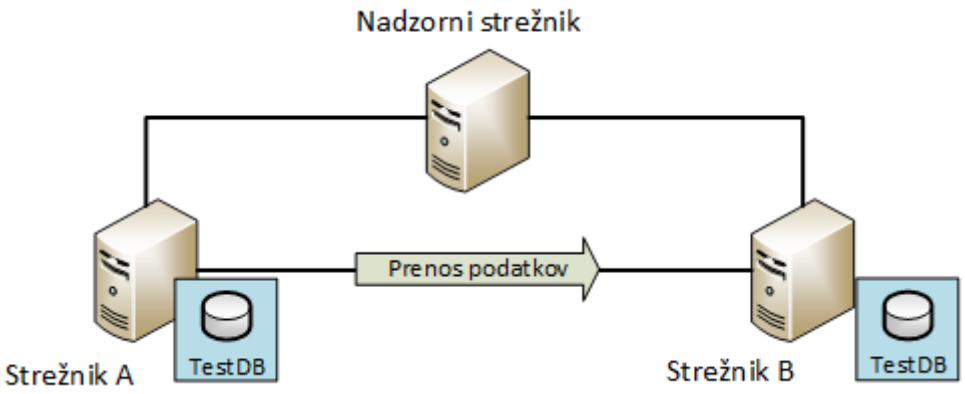


Slika 3.6: Zrcaljenje podatkovne zbirke z visoko varnostjo brez nadzornega strežnika

Visoka varnost delovanja se vzdržuje z naslednjimi koraki:

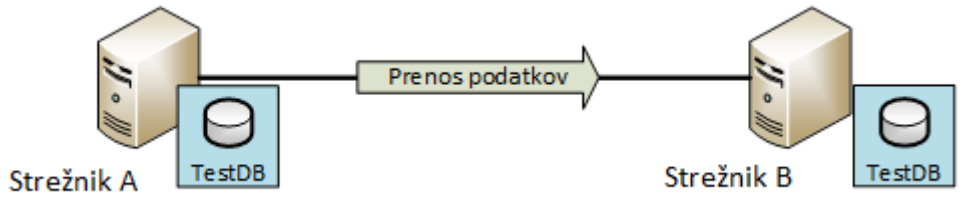
5. Primarni strežnik transakcijo ob prejemu od odjemalca zapiše v transakcijski dnevnik
6. Primarni strežnik transakcijo zapiše v podatkovno zbirko, sočasno pošlje aktivni del transakcijskega dnevnika zrcalnemu strežniku in čaka na potrditev iz zrcalnega strežnika preden odjemalcu sporoči, če je bila transakcija uspešna ali neuspešna
7. Zrcalni strežnik zapiše transakcijski dnevnik v podatkovno zbirko in primarnemu strežniku pošlje potrdilo o vpisu
8. Primarni strežnik pošlje odjemalcu potrditveno sporočilo

Rešitev z visoko varnostjo brez nadzornega strežnika (Slika 3.6) zahteva ročen preklop med primarnim in zrcalnim strežnikom v primeru izpada delovanja primarnega strežnika. Ob prisotnosti nadzornega strežnika (Slika 3.7), rešitev omogoča samodejni preklop med primarnim in zrcalnim strežnikom v primeru izpada delovanja primarnega strežnika. Naloga nadzornega strežnika je preverjanje ali primarni strežnik deluje in zrcalnemu strežniku omogoča izvedbo preklopa med primarnim in zrcalnim strežnikom pri tem pa ne hrani kopije podatkovne zbirke, ki je vključena v zrcaljenje. Samodejen preklop med primarnim in zrcalnim strežnikom se lahko izvede, če zrcalni in nadzorni strežnik izgubi povezavo s primarnim strežnikom in je vzpostavljena povezava med zrcalnim in nadzornim strežnikom. Povezava med zrcalnim in nadzornim strežnikom v primeru izpada primarnega strežnika tvori kvorum, ki ga morata sestavljati vsaj dve instanci strežnika SQL, ki sta vključeni v zrcaljenje podatkovne zbirke, da je podatkovna zbirka lahko dostopna odjemalcem. Pri preklopu postane zrcalna podatkovna zbirka na zrcalnem strežniku primarna podatkovna zbirka in zrcalni strežnik postane primarni strežnik.



Slika 3.7: Zrcaljenje podatkovne zbirke z visoko varnostjo z nadzornim strežnikom

Način delovanja rešitve z visoko zmogljivostjo (Slika 3.8) deluje v asinhronem načinu, ki ne zahteva identično stanje v primarni in zrcalni podatkovni zbirki preden lahko nadaljujemo z delom. Ko pride do transakcije, ki spremeni podatke v podatkovni zbirki na primarnem strežniku se aktivna vsebina transakcijskega dnevnika, ki je povzročila spremembe, pošlje iz primarnega strežnika na zrcalni strežnik, vendar primarni strežnik pri tem ne čaka na zapis sprememb na zrcalnem strežniku. Transakcija se na primarnem strežniku zaključi, ko je le ta vpisana v transakcijski dnevnik in tedaj lahko nadaljujemo z delom. Spremembe se na zrcalni strežnik zapisujejo z zakasnitvijo oz. zaostankom. Ker primarni strežnik ne čaka na potrditev zrcalnega strežnika o uspešnem vpisu sprememb v podatkovno zbirko, ne prihaja do zakasnitev transakcij. Zaostanek zrcalne podatkovne zbirke za primarno podatkovno zbirko je običajno majhen in v praksi znaša nekaj sekund, vendar se lahko poveča, kar je odvisno od obremenitve primarne podatkovne zbirke, hitrosti delovanja zrcalne podatkovne zbirke in hitrosti mrežne ali optične povezave med strežnikoma.



Slika 3.8: Zrcaljenje podatkovne zbirke z visoko zmogljivostjo

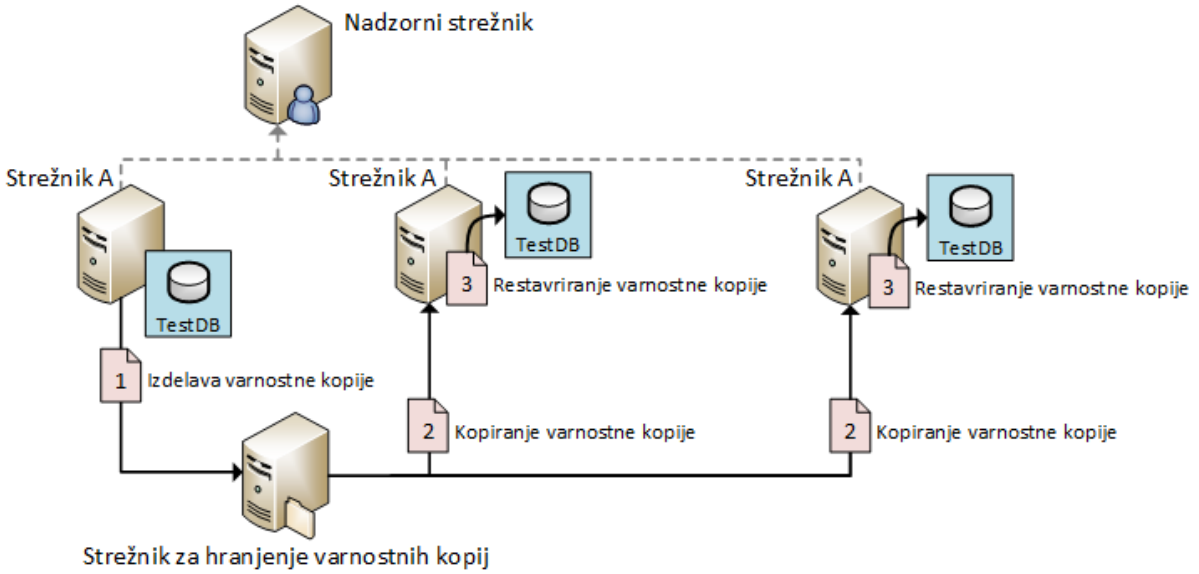
Visoko zanesljivost podatkovne zbirke pri rešitvi z zrcaljenjem podatkovne zbirke zagotavlja način delovanja z visoko varnostjo skupaj z nadzornim strežnikom. Rešitev z načinom delovanja z visoko zanesljivostjo brez nadzornega strežnika je primerna, če je sprejemljiv večji čas izpada, ki je odvisen od hitrosti posredovanja skrbnika sistema, ki lahko opravi ročni

preklop med strežnikoma. Rešitev je enostavna za upravljanje, njena vzpostavitev je enostavna in ne zahteva posebne strojne opreme. Nadzorni strežnik lahko vzpostavimo na enem izmed obstoječih strežnikov, ker za njegovo delovanje zadostuje strežnik SQL Server 2012 Express, ki je brezplačna izdaja. Pri izbiri tovrstne rešitve je potrebno poskrbeti za ročen prenos objektov, ki so povezani s podatkovno zbirko, ki jo vključimo v zrcaljenje (dostopne pravice uporabnikov, povezani strežniki, opravila).

V okviru diplomskega dela smo za postavitev rešitve zrcaljenja podatkovne zbirke uporabili dva strežnika (TEST-SQL01 in TEST-SQL02), ki imata vzpostavljeno medsebojno povezavo z lokalno mrežo in nadzorni strežnik (TEST-DPM). Strežnik TEST-SQL01 služi kot primarna kopija kjer se nahaja podatkovna zbirka TestDB, ki smo jo uporabili v preizkusu. Strežnik TEST-SQL02 služi kot zrcalni strežnik kjer se nahaja zrcalna kopija podatkovne zbirke TestDB. Pri testiranju rešitve visoke zanesljivosti smo izvedli dva načina testiranja z načinom delovanja z visoko varnostjo in samodejnim preklopom. Pri prvem testu smo simulirali preklop med primarnim in zrcalnim strežnikom. V drugem primeru smo izvedli nepričakovano odpoved primarnega strežnika TEST-SQL01. V obeh simulacijah smo beležili čas izpada podatkovnega strežnika in količino izgubljenih podatkov.

3.4 Pošiljanje transakcijskega dnevnika

Pošiljanje transakcijskega dnevnika oz. Log shipping [1, 14, 15] je rešitev, ki deluje na nivoju posamezne podatkovne zbirke in omogoča samodejno repliciranje varnostnih kopij transakcijskega dnevnika podatkovne zbirke iz primarnega strežnika na enega ali več sekundarnih strežnikov, kjer se nahaja replika podatkovne zbirke. Repliciranje varnostnih kopij se izvaja na nivoju posamezne podatkovne zbirke, zato rešitev zahteva ločeno upravljanje vsake podatkovne zbirke, ki jo želimo vključiti v rešitev zagotavljanja visoke zanesljivosti. Za vzpostavitev rešitve potrebujemo najmanj dva strežnika, ki imata vzpostavljeno mrežno povezavo preko katere lahko izmenjujeta podatke, ki so potrebni za prenos podatkov. Opcijsko lahko vključimo tudi dodaten strežnik, ki služi nadziranju vseh ostalih strežnikov, ki so vključeni v pošiljanje transakcijskega dnevnika kateremu strežniki, ki so vključeni v rešitev pošiljajo statistiko delovanja in strežnik kjer se shranjujejo varnostne kopije transakcijskega dnevnika. Slika 3.9 prikazuje primer postavitev rešitve pošiljanja transakcijskega dnevnika z dvema sekundarnima replikama, strežnikom za nadzor in strežnikom za hranjenje varnostnih kopij transakcijskega dnevnika.



Slika 3.9: Primer rešitve pošiljanje transakcijskega dnevnika

Rešitev omogoča omejen bralni dostop pri sekundarni repliki podatkovne zbirke in zaradi zakasnitve pri replikaciji sprememb iz primarne podatkovne zbirke na sekundarno podatkovno zbirko omogoča odpravo napak v primarni podatkovni zbirki, npr. vnos napačnega podatka ali nenameren izbris podatka. Rešitev ne omogoča samodejnega preklopa na sekundarni strežnik v primeru izpada delovanja primarnega strežnika, zahteva ročno

vzdrževanje in repliciranje objektov, ki so povezani s podatkovno zbirko (dostopne pravice uporabnikov, povezani strežniki, opravila). Z naraščanjem števila instanc strežnika SQL Server in števila podatkovnih zbirk postane upravljanje zapleteno in povečuje čas pri preklopu delovanja iz primarne replike na sekundarno repliko. Če pri preklopu iz primarne replike na sekundarno repliko ne izvedemo varnostne kopije transakcijskega dnevnika, lahko pride do izgube podatkov na sekundarni repliki podatkovne zbirke. Količina oz. čas izgube podatkov je odvisen od razpona časa med zadnjo izvedbo varnostne kopije transakcijskega dnevnika in apliciranjem varnostne kopije na sekundarno repliko. V praksi je priporočen čas repliciranja podatkov 600 sekund, zato lahko pride do izgube podatkov v časovnem razponu od 10 (omejitev ponavljanja opravil SQL Server Agent-a) do 600 sekund. V primeru nenačrtovane odpovedi primarnega strežnika je izguba podatkov potrebno sprejemljivo tveganje, ki ga lahko zmanjšamo s krajšim časom repliciranja podatkov. Ker se odjemalci povezujejo na primarni strežnik in povezovalni niz ne podpira parametra v katerem bi lahko navedli ime ali naslov sekundarnega strežnika je potrebno v primeru preklopa na sekundarni strežnik ročno popraviti povezavo do sekundarnega strežnika na vseh odjemalcih kar lahko prinese dodaten čas izpada podatkovnega strežnika.

Pošiljanje transakcijskega dnevnika sestavljajo tri glavne operacije, ki se s pomočjo servisa SQL Server Agent izvajajo samodejno:

1. Izdelava varnostne kopije transakcijskega dnevnika na primarnem strežniku
2. Kopiranje varnostne kopije transakcijskega dnevnika na sekundarni strežnik
3. Restavriranje varnostne kopije transakcijskega dnevnika na sekundarnem strežniku

Rešitev uporablja funkcijo za izdelavo in apliciranje varnostnih kopij podatkovnih zbirk in servis SQL Server Agent, ki je del SQL strežnika oz. Database Engine servisa, zato je rešitev enostavna in zanesljiva. SQL Server Agent skrbi za izvajanje vseh treh operacij, ki sestavljajo pošiljanje transakcijskega dnevnika, v določenem zaporedju z določenim časovnim intervalom. Zaradi uporabe funkcije za izdelavo in apliciranje varnostnih kopij transakcijskega dnevnika je pomembno, da se vse operacije izvajajo neprekinjeno v času delovanja pošiljanja transakcijskega dnevnika na vseh strežnikih, ki so vključeni v pošiljanje transakcijskega dnevnika in v določenem pravilnem zaporedju repliciranja varnostnih kopij transakcijskih dnevnikov.

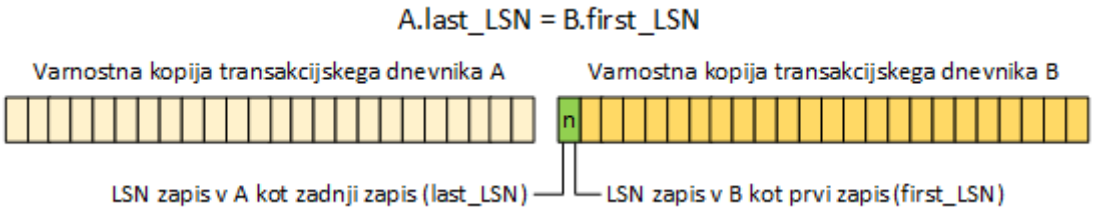
Vsak transakcijski dnevnik [20] vsebuje LSN (Log Sequence Number), ki se deli na FirstLSN in LastLSN. LSN predstavlja zaporedno številko v dnevniku transakcij s pomočjo katere Database Engine tvori pravilno verigo zaporedja transakcij pri restavriranju transakcijskega dnevnika. FirstLSN predstavlja prvi LSN zapis v varnostni kopiji transakcijskega dnevnika,

LastLSN predstavlja zadnji zapis v varnostni kopiji transakcijskega dnevnika. Če imamo LSN2, ki je večji kot LSN1, pomeni, da se je LSN1 zgodil pred LSN2 in je morebiti odvisen od sprememb, ki so se zgodile v LSN1. Če predpostavimo, da imamo varnostno kopijo transakcijskega dnevnika A in B, ki sta med seboj neodvisni in je slednja sledila kopiji A, potem velja, da je LastLSN kopije A enak FirstLSN kopije B kot prikazuje enačba (3.1).

$$A.last\_LSN = B.first\_LSN \tag{3.1}$$

$$A.last\_LSN > B.first\_LSN \tag{3.2}$$

Slika 3.10 prikazuje pravilno LSN zaporedje med dvema varnostnima kopijama. Če je LastLSN kopije A večji od FirstLSN kopije B (3.2) pride do prekrivanja zaporedja, ki onemogoči nadaljevanje restavriranja varnostnih kopij transakcijskega dnevnika na sekundarnih replikah podatkovne zbirke. Če se prekrivanje ne odpravi z ročnim posegom vzdrževalca sistema je potrebna ponovna vzpostavitev rešitve za podatkovno zbirko, pri kateri je prišlo do prekrivanja. Vzrok za prekrivanje je izvedba popolne varnostne kopije podatkovne zbirke ali sprememba lastnosti varnostnega kopiranja podatkovne zbirke.



Slika 3.10: Ustrezno zaporedje LSN oznake v dveh varnostnih kopijah

V okviru diplomskega dela smo za postavitev rešitve pošiljanja transakcijskega dnevnika uporabili dva strežnika (TEST-SQL01 in TEST-SQL02), ki imata vzpostavljeno medsebojno povezavo z lokalno mrežo. Strežnik TEST-SQL01 služi kot primarna replika kjer se nahaja podatkovna zbirka TestDB, ki smo jo uporabili v preizkusu in strežnik kjer se nahajajo varnostne kopije transakcijskih dnevnikov. Strežnik TEST-SQL02 služi kot sekundarna replika, kjer se nahaja replika podatkovne zbirke TestDB kateri repliciramo varnostne kopije transakcijskega dnevnika iz strežnika TEST-SQL01. Varnostne kopije transakcijskega dnevnika smo replicirali s časovnim razmikom 60 sekund. Pri testiranju rešitve visoke zanesljivosti smo izvedli dva načina testiranja. Pri prvem testu smo simulirali preklop med primarnim in sekundarnim strežnikom pri čemer smo pred preklopom izvedli varnostno kopijo transakcijskega dnevnika in jo replicirali na sekundarni strežnik. V drugem primeru

smo izvedli nepričakovano odpoved primarnega strežnika TEST-SQL01. V obeh simulacijah smo beležili čas izpada podatkovnega strežnika in količino izgubljenih podatkov.

4 Primerjava rešitev in rezultatov

4.1 Primerjava lastnosti rešitev visoke zanesljivosti

Ker smo si v okviru diplomskega dela zadali nalogo ugotoviti prednosti in slabosti med posameznimi rešitvami, ki zagotavljajo visoko razpoložljivost podatkovnih zbirk ter oceniti njihovo primernost glede na izbrane kriterije smo pri primerjavi lastnosti rešitev upoštevali konfiguracije posamezne rešitve, ki omogoča najvišjo stopnjo visoke zanesljivosti.

Strežniška gruča:

- Prednosti:
  - Avtonomnost rešitve – hiter samodejen preklop
  - Samodejno delovanje
  - Ni izgube podatkov pri preklopu
  - Deluje na nivoju celotne instance
  - Nizki stroški upravljanja
  - Zahteva malo vzdrževanja
  - Enostavno upravljanje
- Slabosti:
  - Kompleksna implementacija
  - Visok strošek investicije (zaradi potrebe po deljenem diskovnem sistemu)
  - Ne varuje podatkov z replicirano kopijo podatkovne zbirke

Skupina za razpoložljivost:

- Prednosti:
  - Avtonomnost rešitve – hiter samodejen preklop
  - Samodejno delovanje
  - Ni izgube podatkov pri preklopu
  - Varuje podatke z replicirano kopijo podatkovne zbirke
  - Nizki stroški upravljanja
  - Zahteva malo vzdrževanja
  - Enostavno upravljanje
- Slabosti:

- Kompleksna implementacija
- Visok strošek investicije (zaradi potrebe po Enterprise izdaji, ki zahteva količinsko licenciranje)
- Lahko prihaja do zakasnitev pri transakcijah
- Ročen prenos objektov, ki so povezani s podatkovno zbirko (dostopne pravice uporabnikov, povezani strežniki, opravila)

Zrcaljenje podatkovne zbirke:

- Prednosti:
  - Avtonomnost rešitve –samodejen preklop
  - Samodejno delovanje
  - Ni izgube podatkov pri preklopu
  - Varuje podatke z replicirano kopijo podatkovne zbirke
  - Nizki stroški upravljanja
  - Majhen strošek investicije
  - Enostavna implementacija
  - Zahteva malo vzdrževanja
  - Enostavno upravljanje
- Slabosti:
  - Lahko prihaja do zakasnitev pri transakcijah
  - Ročen prenos objektov, ki so povezani s podatkovno zbirko (dostopne pravice uporabnikov, povezani strežniki, opravila)
  - Rešitev je omejena na dva strežnika
  - Potrebuje tretji strežnik v vlogi nadzornega strežnika

Pošiljanje transakcijskega dnevnika:

- Prednosti:
  - Enostavna implementacija
  - Varuje podatke z replicirano kopijo podatkovne zbirke
  - Visoki stroški upravljanja
  - Majhen strošek investicije
  - Zahteva malo vzdrževanja
- Slabosti:
  - Ni samodejnega preklopa
  - Zahtevno upravljanje
  - Izguba podatkov ob nepravilnem postopku preklopa

- Ročen prenos objektov, ki so povezani s podatkovno zbirko (dostopne pravice uporabnikov, povezani strežniki, opravila)

4.2 Testiranje rešitev in primerjava rezultatov

Preizkus delovanja visoke zanesljivosti rešitev s primeri postavitev, ki smo jih opisali v podpoglavjih poglavja Rešitve visoke razpoložljivosti smo razdelili na dva scenarija. V prvem scenariju smo izvedli načrtovani preklop rešitve, ki smo ga izvedli sami s postopkom, ki je predviden za preklop rešitve. V drugem scenariju smo izvedli nenačrtovani preklop rešitve z odklopom mrežne povezave na strežniku kjer je bila v trenutku testiranja rešitev aktivna. Vsak test smo ponovili trikrat in izračunali povprečen čas, ki smo ga dobili pri testiranju posameznega testa.

Pri času preklopa rešitve smo upoštevali čas, ko podatkovna zbirka ni bila dostopna za pisanje aplikaciji SQLSimWrite in ga izračunali z razliko časa med zadnjo transakcijo pred preklopom in prvo transakcijo po. Pri času preklopa rešitve s pošiljanjem transakcijskega dnevnika je bil ključen faktor hitrost izvajanja postopka vzdrževalca pri preklopu med primarnim in sekundarnim strežnikom, kar se kaže z izjemno velikim časom preklopa. Pri količini zgubljenih podatkov smo primerjali število vpisanih vrstic v podatkovno zbirko na posameznem strežniku in poiskali časovne razlike pri neujemanju števila vrstic med posameznima podatkovnima zbirkama. Izjema pri tovrstnem preverjanju je rešitev s strežniško gručo kjer nimamo na voljo kopije podatkovne zbirke s katero bi lahko preverjali vsebino.

Tabela 4.1 prikazuje čas preklopa posamezne rešitve in izgubo podatkov izraženo s časom, pri načrtovanem preklopu rešitve. Tabela 4.2 prikazuje čas preklopa posamezne rešitve in izgubo podatkov izraženo s časom pri nenačrtovanem preklopu rešitve. Pri rešitvi s pošiljanjem transakcijskega dnevnika je prišlo do izgube podatkov v obsegu 26 sekund. Izguba je bila predvidena, ker rešitev ne omogoča varovanja pred izgubo podatkov v primeru nenačrtovanega izpada strežnika, kjer je podatkovna zbirka, ki je vključena v rešitev aktivna.

	Načrtovan preklop (s)	Izgubljeni podatki (s)
Strežniška gruča	12	/
Skupina za razpoložljivost	11	0
Zrcaljenje podatkovne zbirke	4	0
Pošiljanje transakcijskega dnevnika	122	0

Tabela 4.1: Trajanje preklopa pri načrtovanem preklopu

	Nenačrtovan preklop (s)	Izgubljeni podatki (s)
Strežniška gruča	17	/
Skupina za razpoložljivost	32	0
Zrcaljenje podatkovne zbirke	21	0
Pošiljanje transakcijskega dnevnika	87	26

Tabela 4.2: Trajanje preklopa pri nenačrtovanem preklopu

Ker lahko delovanje skupine za razpoložljivost in zrcaljenje podatkovne zbirke v sinhronem načinu povzroči zakasnitev pri izvajanju transakcij smo izmerili razliko v številu transakcij med sinhronim in asinhronim načinom delovanja. Vsak test smo ponovili trikrat s časom izvajanja 10 sekund in nato še trikrat s časom izvajanja 60 sekund. Za vsak način delovanja za določeno rešitev smo izračunali povprečno zaokroženo vrednost zapisanih transakcij, ki smo jih predstavili v Tabela 4.3 in Tabela 4.4.

	Skupina za razpoložljivost	Zrcaljenje podatkovne zbirke
Sinhron način delovanja	6011	4125
Asinhron način delovanja	7424	7153

Tabela 4.3: Število transakcij v 10 sekundah pri asinhronem in sinhronem načinu delovanja

	Skupina za razpoložljivost	Zrcaljenje podatkovne zbirke
Sinhron način delovanja	41807	30072
Asinhron način delovanja	64851	60369

Tabela 4.4: Število transakcij v 60 sekundah pri asinhronem in sinhronem načinu delovanja

5 Sklep

Cilj diplomskega dela je bilo analizirati rešitve, ki zagotavljajo visoko razpoložljivost podatkovnih zbirk strežnika Microsoft SQL Server 2012. Cilj analize je bilo analiziranje posameznih rešitev s pomočjo aplikacije, ki smo jo razvili za pomoč pri testiranju in pri tem ugotoviti njihovo zanesljivost, avtonomnost in hitrost delovanja, sposobnost varovanja podatkov ter določiti stroške upravljanja in investicije, kar bi nam olajšalo odločanje pri izbiri rešitve med načrtovanjem infrastrukturnih rešitev.

V času izdelave te diplomske naloge smo podrobno preučili delovanje posameznih rešitev, ki jih podpira strežnik Microsoft SQL Server 2012 in tako spoznali njihove poglavitne lastnosti, področje uporabnosti, prednosti in slabosti, ki smo jih lahko na koncu preizkusili in preverili pri implementiranju. Sama implementacija rešitev je v testno pripravljenem okolju potekala z nekaj zapleti kjer se je izkazalo, da določene rešitve zahtevajo naprednejše poznavanje programske opreme in tehnologije, ki smo jo uporabili. Pri implementaciji strežniške gruče je prišlo do težav pri nameščanju komponent strežnika SQL Server in pri implementaciji skupine za razpoložljivost je prišlo do težav zaradi nepravilno konfiguriranih pravic v domenskem imeniku. Vse težave smo sicer na podlagi predhodnih izkušenj relativno hitro odpravili. Pri implementiranju zrcaljenja podatkovne zbirke in pošiljanja transakcijskega dnevnika nismo imeli težav.

Kljub težavam pri implementaciji strežniške gruče in skupine za razpoložljivost sta se obe rešitvi izkazali z enostavnim upravljanjem in zanesljivim delovanjem. Uporaba obeh rešitev je v času testiranja delovala brezhibno in brez težav. Tudi rešitev zrcaljenje podatkovne zbirke je v času testiranja delovala zanesljivo in brez težav, pri tem pa nismo imeli težav z upravljanjem. Rešitev s pošiljanjem transakcijskega dnevnika je sicer enostavna za implementacijo, vendar je njeno upravljanje zahtevno in zahteva izkušenost pri upravljanju saj v nasprotnem primeru lahko hitro pride do izgube podatkov in razpoložljivosti podatkovne zbirke za uporabo v produkciji. V času testiranja smo morali večkrat ponoviti implementacijo rešitve, ker smo zaradi kompleksnosti upravljanja in napačnega postopka onemogočili repliciranje podatkov med primarno in sekundarno kopijo podatkovne zbirke.

Kljub zahtevnejši implementaciji strežniške gruče in skupine za razpoložljivost se rešitvi izkažeta za zelo uporabni in s tem priporočljivi izbiri pri načrtovanju infrastrukturnih rešitev.

Zakasnitev transakcij v sinhronem načinu delovanja skupine za razpoložljivost se je izkazala za nekoliko počasnejšo rešitev, zato je priporočljivejša izbira strežniška gruča. Slabost obeh rešitev je povezana s stroškom investicije, kjer strežniška gruča zahteva investicijo v omrežno diskovno polje, skupina za razpoložljivost pa zahteva Enterprise izdajo strežnika SQL Server, katera je na voljo samo v okviru količinskega licenciranja, ki predstavlja dodaten strošek, ki lahko preseže investicijo v omrežno diskovno polje, ki ga danes že pogosto srečamo v produkcijskem okolju.

Razvoj aplikacije SQLSimWrite je poenostavil preizkus implementiranih rešitev in odprl nove ideje za razširitev in izboljšavo aplikacije. Poleg osnovnih funkcionalnosti, ki smo jih implementirali za potrebe diplomskega dela bi aplikacijo lahko razširili z izbiro različnih gonilnikov za povezovanje s strežnikom SQL, nastavljanjem parametrov povezave kot so čas do ponovne ponovitve povezave in število ponovitev, izbiro za večje število sočasnih povezav, ki se vzpostavijo s strežnikom SQL in s tem večje število sočasnih transakcij, ločevanje posamezne transakcije v okviru zapisanih podatkov v podatkovni zbirki, primerjava zakasnitve transakcije med odjemalcem in strežnikom in analiza vpisanih podatkov s strani aplikacije, ki bi nam lahko postregla s podatki o času izpada strežnika SQL in količini izgubljenih podatkov.

Analizo bi lahko ocenil kot uspešno, saj je zahtevala podrobno preučitev delovanja strežnika SQL Server 2012 in njegovih funkcionalnosti, kar bo pripomoglo pri bodočem odločanju in načrtovanju infrastrukturnih rešitev s katerimi se bomo soočili.

Literatura

[1] H. S. Goswami, *Microsoft SQL Server 2008 High Availability*, Birmingham: Packt Publishing, 2011, str. 13-83, 203-253.

[2] P. LeBlanc, *Microsoft SQL Server 2012 Step by Step*, California: O'Reilly Media, 2013, str. 1-432.

[3] K. Schmidt, *High Availability and Disaster Recovery*, Frankfurt: Springer Berlin Heidelberg, 2006, str. 108-125, 199-205.

[4] J. Sack, S. Mishra (junij 2012). *AlwaysOn Architecture Guide: Building a High Availability and Disaster Recovery Solution by Using Failover Cluster Instances and Availability Groups*. [Online]. Dostopno: http://download.microsoft.com/download/D/2/0/D20E1C5F-72EA-4505-9F26-FEF9550EFD44/Building_a_HA_and_DR_Solution_using_AlwaysON_SQL_FCIs_and_AGs%20v1.docx

Spletni viri:

[5] M. Perlin. (september 2012). Downtime, Outages and Failures - Understanding Their True Costs. *Evolven.* [Online]. Dostopno: http://www.evolven.com/blog/downtime-outages-and-failures-understanding-their-true-costs.html

[6] Datacenter Dynamics. (december 2013) One minute of data center downtime costs US$7,900 on avarage. *Datacenter Dynamics.* [Online]. Dostopno: http://www.datacenterdynamics.com/focus/archive/2013/12/one-minute-data-center-downtime-costs-us7900-average

[7] Pingdom. (avgust 2009). The PayPal outage cost its users between 7 and 32 million USD. *Pingdom* [Online]. Dostopno: http://royal.pingdom.com/2009/08/04/the-paypal-outage-cost-its-users-between-7-and-32-million-usd/

[8] (2014) MSDN: C# Programming Guide. Dostopno: http://msdn.microsoft.com/en-us/library/67ef8sbd.aspx

[9] (2014) MSDN: High Availability Solutions (SQL Server). Dostopno: http://msdn.microsoft.com/en-us/library/ms190202(v=sql.110).aspx

[10] (2014) MSDN: Introduction to the C# Language and the .NET Framework. Dostopno: http://msdn.microsoft.com/en-us/library/z1zx9t92.aspx

[11] (2013) Technet: Announcing the General Availability of Windows Server 2012 R2: The Heart of Cloud OS. Dostopno: http://blogs.technet.com/b/windowsserver/archive/2013/10/18/announcing-the-general-availability-of-windows-server-2012-r2-the-heart-of-cloud-os.aspx

[12] (2014) Technet: Failover Clusters. Dostopno: http://technet.microsoft.com/en-us/library/cc754482.aspx

[13] (2012) Technet: Failover Clustering Overview. Dostopno: http://technet.microsoft.com/en-us/library/hh831579.aspx

[14] (2014) Technet: High Availability (Database Engine). Dostopno: http://technet.microsoft.com/en-us/library/bb522583(v=sql.105).aspx

[15] (2014) Technet: High Availability Solutions (SQL Server). Dostopno: http://technet.microsoft.com/en-us/library/ms190202(v=sql.110).aspx

[16] (2014) Technet: Microsoft HPC Pack (Windows HPC Server). Dostopno: http://technet.microsoft.com/en-us/library/cc514029.aspx

[17] (2014) Technet: Understanding Database Availability Groups. Dostopno: http://technet.microsoft.com/en-us/library/dd979799(v=exchg.141).aspx

[18] (2014) Technet: Windows Server 2012 R2 and Windows Server 2012. Dostopno: http://technet.microsoft.com/en-us/library/hh801901.aspx

[19] (2014) Technet: Welcome to Visual Studio 2013. Dostopno: http://msdn.microsoft.com/en-us/library/dd831853.aspx

[20] (2014) Technet: Working with Restore Sequences for SQL Server Databases. Dostopno: http://technet.microsoft.com/en-us/library/ms187486(v=sql.105).aspx

[21] (2014) Wikipedia: Microsoft SQL Server. Dostopno: http://en.wikipedia.org/wiki/Microsoft_SQL_Server